

Project Zipline Prefix Attach

Micro Architecture Specification

**Authors:**

**Microsoft Corporation**

**Broadcom Corporation**

**Revision History**

|  |  |
| --- | --- |
| Date | Description |
| ??/??/2019 | Version 1.0 |
|  |  |

**License**

Contributions to this Specification are made under the terms and conditions set forth in the Open Web Foundation Contributor License Agreement (“OWF CLA 1.0”) (“Contribution License”) by:

Microsoft Corporation

Broadcom Corporation

Usage of this Specification is governed by the terms and conditions set forth in Open Web Foundation Final Specification Agreement (“OWFa 1.0”) (“Specification License”).

**Note**: The following clarifications, which distinguish technology licensed in the Contribution License and/or Specification License from those technologies merely referenced (but not licensed), were accepted by the Incubation Committee of the OCP:

**None.**

NOTWITHSTANDING THE FOREGOING LICENSES, THIS SPECIFICATION IS PROVIDED BY OCP "AS IS" AND OCP EXPRESSLY DISCLAIMS ANY WARRANTIES (EXPRESS, IMPLIED, OR OTHERWISE), INCLUDING IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR A PARTICULAR PURPOSE, OR TITLE, RELATED TO THE SPECIFICATION. NOTICE IS HEREBY GIVEN, THAT OTHER RIGHTS NOT GRANTED AS SET FORTH ABOVE, INCLUDING WITHOUT LIMITATION, RIGHTS OF THIRD PARTIES WHO DID NOT EXECUTE THE ABOVE LICENSES, MAY BE IMPLICATED BY THE IMPLEMENTATION OF OR COMPLIANCE WITH THIS SPECIFICATION. OCP IS NOT RESPONSIBLE FOR IDENTIFYING RIGHTS FOR WHICH A LICENSE MAY BE REQUIRED IN ORDER TO IMPLEMENT THIS SPECIFICATION. THE ENTIRE RISK AS TO IMPLEMENTING OR OTHERWISE USING THE SPECIFICATION IS ASSUMED BY YOU. IN NO EVENT WILL OCP BE LIABLE TO YOU FOR ANY MONETARY DAMAGES WITH RESPECT TO ANY CLAIMS RELATED TO, OR ARISING OUT OF YOUR USE OF THIS SPECIFICATION, INCLUDING BUT NOT LIMITED TO ANY LIABILITY FOR LOST PROFITS OR ANY CONSEQUENTIAL, INCIDENTAL, INDIRECT, SPECIAL OR PUNITIVE DAMAGES OF ANY CHARACTER FROM ANY CAUSES OF ACTION OF ANY KIND WITH RESPECT TO THIS SPECIFICATION, WHETHER BASED ON BREACH OF CONTRACT, TORT (INCLUDING NEGLIGENCE), OR OTHERWISE, AND EVEN IF OCP HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Contents

[1 Overview 6](#_Toc3895996)

[1.1 Prefix Attach Block Diagram 6](#_Toc3895997)

[1.2 Interfaces 6](#_Toc3895998)

[1.3 Data Flow 8](#_Toc3895999)

[1.3.1 Inbound Data Flow 9](#_Toc3896000)

[1.3.2 Outbound Data Flow 9](#_Toc3896001)

[1.3.3 Control Data Flow 9](#_Toc3896002)

[1.3.4 Frame Header Modifications 10](#_Toc3896003)

[2 Detailed Description 11](#_Toc3896004)

[2.1 Prefix Data SRAM 11](#_Toc3896005)

[2.2 Predetermined Huffman Table SRAM 11](#_Toc3896006)

[2.3 Prefix Attach Inbound Controller 11](#_Toc3896007)

[2.4 Outbound Controller 12](#_Toc3896008)

[3 Debug 13](#_Toc3896009)

[3.1 Control Status Registers (CSR) 13](#_Toc3896010)

Table of Figures

[Figure 1 Prefix Attach Block Diagrm 6](#_Toc3895980)

[Figure 2 Top level block diagram for the CCEIP 9](#_Toc3895981)

[Figure 3 Prefix Attach frame stack up 10](#_Toc3895982)

Table of Tables

# Overview

The Prefix Attach, when enabled, examines the incoming data to determine the pre-existing prefix data that should be used to seed the compression engine to achieve the best compression.

## Prefix Attach Block Diagram



Figure Prefix Attach Block Diagrm

## Interfaces

The Prefix Engine external interfaces are shown below.

| Name | I/O | Description |
| --- | --- | --- |
| **Clocks and resets** |  |  |
| clk | I | 800MHz clock |
| rst | I | Active high reset |
| **AXI4S Slave** |  |  |
| tvalid | I | TVALID indicates that the master is driving a valid transfer.  A transfer takes place when both TVALID and TREADY are asserted. |
| tlast | I | TLAST indicates the boundary of a packet. |
| tid[13:0] | I | TID is the data stream identifier that indicates different streams of data. |
| tkeep[7:0] | I | TKEEP is the byte qualifier that indicates whether the content  of the associated byte of TDATA is processed as part of the data stream.  Associated bytes that have the TKEEP byte qualifier deasserted are null bytes and can be removed from the data stream. |
| tuser[7:0] | I | TUSER is user defined sideband information that can be transmitted alongside the data stream. |
| tdata[63:0] | I | TDATA is the primary payload that is used to provide the data that is passing across the interface. The width of the data payload is an integer number of bytes. |
| tready | O | TREADY indicates that the slave can accept a transfer in the current cycle. |
| **AXI4S Master** |  |  |
| tvalid | O | TVALID indicates that the master is driving a valid transfer.  A transfer takes place when both TVALID and TREADY are asserted. |
| tlast | O | TLAST indicates the boundary of a packet. |
| tid[13:0] | O | TID is the data stream identifier that indicates different streams of data. |
| tkeep[7:0] | O | TKEEP is the byte qualifier that indicates whether the content  of the associated byte of TDATA is processed as part of the data stream.  Associated bytes that have the TKEEP byte qualifier deasserted are null bytes and can be removed from the data stream. |
| tuser[7:0] | O | TUSER is user defined sideband information that can be transmitted alongside the data stream. |
| tdata[63:0] | O | TDATA is the primary payload that is used to provide the data that is passing across the interface. The width of the data payload is an integer number of bytes. |
| tready | I | TREADY indicates that the slave can accept a transfer in the current cycle. |
| **RBUS In** |  |  |
| addr[31:0] | I | Rbus byte address |
| wr\_strb | I | Rbus write strobe indicates start of a write transaction |
| wr\_data[31:0] | I | Rbus write data |
| rd\_strb | I | Rbus read strobe indicates the start of a read transaction |
| rd\_data[31:0] | I | Rbus read data |
| ack | O | Rbus acknowledge indicates the end of a transaction |
| err\_ack | O | Rbus error acknowledge indicates an aborted transaction due to error |
| **RBUS Out** |  |  |
| addr[31:0] | O | Rbus byte address |
| wr\_strb | O | Rbus write strobe indicates start of a write transaction |
| wr\_data[31:0] | O | Rbus write data |
| rd\_strb | O | Rbus read strobe indicates the start of a read transaction |
| rd\_data[31:0] | O | Rbus read data |
| ack | I | Rbus acknowledge indicates the end of a transaction |
| err\_ack | I | Rbus error acknowledge indicates an aborted transaction due to error |

## Data Flow

Below is a top level block diagram for the CCEIP, with the Prefix Attach highlighted in blue, to show how it fits into the overall Project Zipline data flow:



Figure Top level block diagram for the CCEIP

### Inbound Data Flow

The inbound Data Frames are presented to the Prefix Attach Engine via the Streaming AXI Slave interface. The data frames are written into a TLV Parser. The TLV Parser is a common block that can be instantiated in any Project Zipline Engine sub-design. The TLV Parser is used to split out the inbound TLV data stream into TLV’s of interest to the block’s main logic and those which the block doesn’t user are simply passed through. Additionally, the TLV Parser will reassemble the two data streams back into one outbound AXI stream.

### Outbound Data Flow

The AXI Master Streaming I/F streams out the Frame data, when the downstream device indicates it is ready for data. The inbound Compression Header contains a Prefix mode and Prefix Selection field. The Prefix Attach engine uses this to optionally add the Prefix Header and the Predetermined Huffman Header to the outbound data stream.

### Control Data Flow

A Rbus Ring Interface will be provided to provide for Configuration & Status Register (CSR) access as well as various debug facilities. The registers provided by this interface will provide read/write access to the following:

* AXI streaming interface statistics, i.e. data counters
* Loading of the Prefix Data SRAM
* Loading of the Predetermined Huffman Data SRAM
* Debug facilities

### Frame Header Modifications

Refer to the Project Zipline Document for a description of each Project Zipline TLV Header.



Figure Prefix Attach frame stack up

# Detailed Description

## Prefix Data SRAM

The Prefix Data SRAM is loaded at configuration time with 64 predetermined XP10 Prefix headers. Each header occupies 130 locations organized as a header word, 128 data words and a precalculated CRC word. The CRC will be calculated as the header is read and compared with the stored CRC value. A CRC error will cause the Prefix Attach to modify the Footer TLV to indicate the error.

## Predetermined Huffman Table SRAM

The Predetermined Huffman Table SRAM is loaded at configuration time with 64 predetermined Huffman headers. Each header occupies 130 locations organized as a header word, 128 data words and a precalculated CRC word. The CRC will be calculated as the header is read and compared with the stored CRC value. A CRC error will cause the Prefix Attach to modify the Footer TLV to indicate the error.

## Prefix Attach Inbound Controller

The Prefix Attach DMA state machine controls the flow of frame data from the input AXI streaming interface to the output AXI Streaming interface. When inbound data is available from the TLV Parser, the Inbound Controller will read out the frames and extract the Prefix mode and Prefix Selector from the Compression Header. Inbound data will be processed as follows:

* No Prefix (Prefix Mode = 0)

The inbound data is passed to the outbound data stream with no prefix data added.

* Prefix (Prefix Mode = 1)

The Prefix Selector is used as an index into the Prefix Data SRAM and 1 kB of data is added to the outbound data stream.

* Prefix with Huffman (Prefix Mode = 2)

The Prefix Selector is used as an index into the Prefix Data SRAM and as an index into the Predetermined Huffman Table Data SRAM. The DMA state machine will add 1 kB of data from the Predetermined Huffman Table Data SRAM, followed by 1kB of data from the Prefix Data SRAM to the outbound data stream.

* User Prefix (Prefix Mode = 3)

In User Prefix mode the inbound data is passed to the outbound data stream with no prefix data added same as Prefix Mode=0. In User Prefix mode the prefix data is already present in the inbound data stream.

## Outbound Controller

The Inbound Controller will communicate to the Outbound Controller if a Prefix and optionally a Predetermined Huffman TLV is to inserted. The Outbound Controller reads the data from the appropriate SRAM and formats it as a valid TLV. This TLV is presented to the TLV Parser as a new TLV to be inserted into the outgoing data stream.

# Debug

## Control Status Registers (CSR)

The Prefix engine will include Control Status Registers (CSR) accessible via the Rbus interface to facilitate Debug/Monitor of the Prefix Engine. These CSR’s shall include the following:

1. TLV Parser configuration
2. Access to read/write the internal SRAMs